DDNS for Non-compliant Dual-stack Clients Solution Proposal for ISC DHCP

Introduction:

RFC 4703, Section 5.2 states that:

“Sites that wish to permit a single FQDN to contain both A and AAAA RRs MUST make use of DHCPv4 clients and servers that support using the DHCP Unique Identifier (DUID) for DHCPv4 client identifiers such that this DUID is used in computing the RDATA of the DHCID RR by both DHCPv4 and DHCPv6 for the client;”

While ISC DHCP server and client both support RFC 4703, there are any number of alternative DHCP clients that do not. Without this compliance, there is no concrete mechanism for DHCP servers to positively and uniquely attribute DHCPv4 and DHCPv6 packets as emanating from the same client, and without DHCP servers and clients adhering to the same rules regarding DDNS updates, performing DDNS in under such conditions is problematic at best.

ISC DHCP does not currently support any combination of configuration options that would allow for the reliable participation in DDNS of non-compliant dual-stack clients. This document proposes a solution by which ISC DHCP server can accommodate such clients.

General:

The proposed solution will introduce a new DDNS conflict resolution mode, Dual-Stack Mixed Mode (**DSMM**), which can be enabled through a new configuration parameter, **ddns-dual-stack-mixed-mode.** DSMM operates under the premise that all DHCP servers supporting one protocol (v4 or v6) will use one DDNS update style (interim or standard) while all servers supporting the “other” protocol use the “other” update style. This ensures that the type of guard record (TXT or DHCID) is always the same for a given protocol.

In short, when DSMM is enabled a DHCP server will ignore DNS address and guard records of the “other” type when deciding if updates can be made or whether the existing entries are either static or belong to another client and must be left alone. In no case, will address or guard records of the “other” type be altered by a DSMM server.

DSMM behavior will be modifiable through two additional configuration parameters. The first, **ddns-other-guard-is-dynamic**, if enabled instructs the server to treat cases in which the only guard record present is of the "other" guard type as dynamic, regardless of whether or not an address record is present. This parameter is off by default.

The second modifier, **ddns-guard-id-must-match**, enables or disables the requirement that a guard record's data value must match the client ID requesting the update. In other words, the mere presence of the guard record indicates a dynamic entry which may be updated. This permits dynamic clients to overwrite each other's entries while protecting static entries.

The criteria for a dynamic entry that may be updated can be expressed as follows:

1. No records for this FQDN exist
2. An address record is present and the guard record, optionally matching the client ID, is present
3. An address record is not present but the guard record, optionally matching the client ID is present
4. Optionally, the only guard record present is the other type of guard record, regardless of whether or not the address record is present

Deployers should bear in mind that DSMM has limitations and that sites where DSMM is in use must configure its servers uniformly and that allowing updates from clients should be avoided. Note that DSMM does not apply to updating Reverse DNS.

Details:

ISC DHCP updates the DNS mapping for an FQDN by issuing a series of one or more DDNS update requests, depending upon the entries (or lack thereof) found in DNS for the FQDN in question. Each request consists of a set of conditions or prerequisites and a set of DNS record updates. The prerequisites express what must be true about the existing DNS state for the FQDN in order for the set of record updates to be applied. There is a one series of update requests used to add a mapping of an FQDN to an address, and another for removing that mapping. Adding a mapping will be discussed first.

Forward DNS Add/Replace:

When update-conflict-detection is enabled, the current logic for adding Forward DNS entries consists of two update requests. The first request, ADD1, adds an address and guard records only if no entries (of any type) for the FQDN already exist. If this succeeds the second request is not needed. However, if it fails because one or more entries for the FQDN already exist, the second request, ADD2, is issued.

ADD2 requires the presence of a guard record with a matching client ID exist. If this condition is true then the existing address is replaced. These requests expressed in nsupdate syntax are shown below:

ADD1:
```
prereq nxdomain <name>                      # no records of any kind for this FQDN
update add <name> <addr_t> <address>        # adds the new address record
update add <name> <guard_t> <client-id>     # adds the new guard record
```

If ADD1 fails with YXDOMAIN (i.e. name is in use), then attempt ADD2

ADD2:
```
prereq yxrrset <name> <guard_t> <client-id> # guard with matching client ID exists
update delete <name> <addr_t>               # delete existing address record
update add <name> <addr_t> <address>        # add the new address record
```

In order to implement DSMM, a series of three update requests will be used. The first request, ADD1, will be the same as above. It will create the address and guard record only if no entries for the FQDN exist.

The second request, ADD2, will be the same unless client-id matching is disabled.   In that case, it will contain a prerequisite that the guard record exists followed by record updates to first delete the existing address and guard and then add the new address and guard.

ADD2-ID-OFF (ddns-guard-id-must-match off):
```
prereq yxrrset <name> <guard_t>           # guard must exist
update delete <name> <addr_t>             # delete existing address record
update delete <name> <guard_t>            # delete existing guard record
update add <name> <addr_t> <address>      # add the new address record
update add <name> <guard_t> <client-id> # add new guard record
```

When DSMM is disabled and ADD2 fails nothing more is done. In a dual-stack environment, this leads to issues where the presence of "other" record types is not allowing updates to occur.   For example, assume a DHPCv4 server using the interim update style attempts to add an FQDN for which an AAAA address record and a DHCID guard record already exist. ADD1 one will fail because the name is in use and ADD2 will fail because the TXT guard record does not exist.   In order to address this scenario, enabling DSMM adds a third update request, ADD3.

There will be two variants of ADD3. The first variant is used by default with DSMM. It consists of prerequisites which require that neither an address nor guard record of "my" type exist but that a guard record of the "other" type does exist, followed by updates to add the address and guard record:

ADD3:
```
prereq nxrrset <name> <addr_t>            # no address record of my type
prereq nxrrset <name> <guard_t>           # no guard record of my type
prereq yxrrset <name> <other_guard_t>     # other guard type does exist
update add <name> <addr_t> <address>      # add the new address record
update add <name> <guard_t> <client-id>   # add the new guard record
```

The second variant, ADD3-OTHER, will be used with DSMM when **ddns-other-guard-is-dynamic** is enabled. This variant removes the requirement that no address record of "my" type exists and adds an update to delete the address record (in case it does exist). It is intended to permit updates to address records which would otherwise be interpreted as static (no accompanying guard of its type) because a guard of the other type does exist. Under default handling, this would be considered a static entry and the update would not be allowed. Use of ddns-other-guard-is-dynamic essentially states that the presence of the other type of guard indicates that entries for this FQDN of either type are dynamic.

ADD3-OTHER (ddns-other-guard-is-dynamic on):
```
prereq yxrrset <name> <other_guard_t>     # other guard type exists
update delete <name> <addr_t>             # delete existing address record (if one)
update add <name> <addr_t> <address>      # add new address record
update add <name> <guard_t> <client-id>   # add new guard record
```

Forward DNS Remove:

Deleting forward entries is currently done using two update requests. The first request, REM1, deletes an FQDN's address record for a given address only if a guard record matching the client id exists. This request is always used whether conflict detection is on or not. This is done to protect static entires. REM1 is shown in nsupdate syntax below:

REM1:
```
prereq yxrrset <name> <guard_t> client-id # guard with matching client id exists
update delete <name> <addr_t> <address>   # delete the address record
```

The implementation of client-id matching will alter REM1 when dns-guard-id-must-match is off by omitting the client-id from the guard record prerequisite. Thus alteing the condition to require only that the guard record exist.

If REM1 fails then then DNS entry is presumed to be either a static entry (no guard at all or that it belongs to another client (guard is present but does not match) and no further action is taken. If it succeeds however, a second request, REM2, is issued to delete the guard record if there are no other address records of either type. REM2 is shown below:

REM2:
```
prereq nxrrset name <addr_t>       # no records of this address type
prereq nxrrset name <other_addr_t> # no records of other address type
update delete name <guard_t> <client-id>  # delete the guard record
```

The DMSS solution will introduce the following variations. REM1 will be same as the above unless client-id matching is disabled (i.e. ddns-guard-id-must-match off). In that case the guard record prerequisite will omit the client-id, thus altering it to require only that the guard record exist. This variant is shown below:

REM1-ID-OFF (ddns-guard-id-must-match off)**:**
```
prereq yxrrset <name> <guard_t>         # guard record exists
update delete <name> <addr_t> <address> # delete the address record
```

When DSMM is enabled and REM1 succeeds then the second request issued will be REM2-DMSS. This request will contain a single prerequisite requiring there be no more address records of "my" type and a single update to delete of the guard record. It is shown below:

REM2-DSMM:
```
prereq nxrrset name <addr_t>  # no records of this address type exist
update delete name <guard_t>  # delete the existing guard record
```

As stated earlier, normally the failure of REM1 indicates no further action should be taken, If however REM1 fails when DSMM is enabled and the server is configured to consider the presence of the other type guard record as indicting a dynamic entry (i.e. ddns-other-guard-is-dynamic on) then a second remove request, REM2-DSMM-OTHER, will be issued. It will delete the address record if the only guard record that exists is of the "other" guard type. This remove is shown below:

REM2-DSMM-OTHER (ddns-other-guard-is-dynamic on):
```
prereq nxrrset <name> <guard_t>        # my guard type does not exist
prereq yxrrset <name> <other_guard_t>  # other guard type does exist
update delete <name> <addr_t> address  # delete the existing address record
```

This request is intended to permit deleting an address record, which would otherwise be interpreted as static (no accompanying guard of its type), when only a guard of the other type exists:

Truth Table:

Attached is a truth table which details when updates/deletes should or should not be done when ddns-dual-stack-mixed-mode is enabled, and when client id matching is both on and off. The table is written from the perspective of single server running either protocol as the logic is same. Address type or refers to either A or AAAA, guard type refers to either TXT of DHCID. The adjective "my" means the type of record the server is using while "other" means the type of record used by the servers running the other protocol.

The columns are:

other address type - 0 no record, 1 record present
my address type - 0 no record,1 record present
other guard type - 0 no record, 1 record present
my guard type - 0 no record, 1 record present
this client – whether the data value of guard record matches the update client id: 0 = does not match, 1 = does match
description - text description of the condition

The remaining columns describe what, if any changes will be made to the DNS data and which update will make them.

Other address type	My address type	Other guard type	My guard type	This client	description	What happens on DSMM Add	What happens on DMSS Add with client-id matching off	What happens on DSMM Remove	What happens on DSMM Remove with client-id matching off
0	0	0	0	0	no entries at all	Adds address and guard (ADD1)	Adds address and guard (ADD1)	Nothing	Nothing
0	0	0	0	1	can't happen				
0	0	0	1	0	my guard type but not this client (guard RR has been orphaned)	Nothing	Adds address, replaces guard (ADD2-ID-OFF)	Nothing	Deletes guard (REM1-ID-OFF, REM2-DSMM)
0	0	0	1	1	my guard type and this client (our guard RR is orphaned)	Adds address, replaces guard. (ADD2)	Adds address, replaces guard, (ADD2-ID-OFF)	Deletes guard (REM1,REM2-DSMM)	Deletes guard (REM1-ID-OFF,REM2-DSMM)
0	0	1	0	0	other guard only … (and the other guard RR is orphaned, but we don't care about cleaning it up - that's the other server's problem)	Add address, add guard, (ADD3)	add address, add guard, (ADD3)	Nothing	Nothing
0	0	1	0	1	can't happen				
0	0	1	1	0	other guard and my guard type but not this client (both these guard RRs are orphaned)	Nothing	Adds address, replaces guard, (ADD2-ID-OFF)	Nothing	Deletes guard (REM1-ID-OFF,REM2-DSMM)
0	0	1	1	1	other guard and my guard type and this client (both guard RRs are orphaned)	Adds address, replaces guard, (ADD2)	Adds address, replaces guard, (ADD2-ID-OFF)	Deletes guard, (REM1,REM2-DSMM)	Deletes guard (REM1-ID-OFF,REM2-DSMM)
0	1	0	0	0	address only …. static	Nothing	Nothing	Nothing	Nothing
0	1	0	0	1	can’t happen				
0	1	0	1	0	conflict - address owned by other	Nothing	Replaces address, replaces guard, (ADD2-ID-OFF)	Nothing	Deletes address and guard (REM1-ID-OFF,REM2-DSMM)
0	1	0	1	1	my entry	Replaces address, replaces guard, (ADD2)	Replaces address, replaces guard, (ADD2-	Deletes address and guard (REM1,REM2-	Deletes address and guard (REM1-ID-OFF,REM2-

Other address type	My address type	Other guard type	My guard type	This client	description	What happens on DSMM Add	What happens on DMSS Add with client-id matching off	What happens on DSMM Remove	What happens on DSMM Remove with client-id matching off
							ID-OFF)	DSMM)	DSMM)
0	1	1	0	0	my address but other guard (the other guard is orphaned)	Nothing UNLESS **ddns-other-guard-is-dynamic** ON: then replace address, add guard, (ADD3-OTHER)	Nothing UNLESS **ddns-other-guard-is-dynamic** ON: then replace address, add guard, (ADD3-OTHER)	Nothing UNLESS **ddns-other-guard-is-dynamic** ON: then delete address (REM1,REM2-DSMM-OTHER)	Nothing, UNLESS **ddns-other-guard-is-dynamic** ON: then delete address (REM1-ID-OFF, REM2-DSMM-OTHER)
0	1	1	0	1	can’t happen				
0	1	1	1	0	conflict - address owned by other	Nothing	Replaces address, replaces guard, (ADD2-ID-OFF)	Nothing	Deletes address and guard
0	1	1	1	1	my entry	Replaces address, replaces guard, (ADD2)	Replaces address, replaces guard, (ADD2-ID-OFF)	Deletes address and guard (REM1, REM2-DSMM)	Deletes address and guard (REM1-ID-OFF,REM2-DSMM)
1	0	0	0	0	other address only … static	Nothing	Nothing	Nothing	Nothing
1	0	0	0	1	can’t happen				
1	0	0	1	0	my guard type but not this client (orphaned guard RR)	Nothing	Adds address, replaces guard, (ADD2-ID-OFF)	Nothing	Deletes guard (REM1-ID-OFF,REM2-DSMM)
1	0	0	1	1	my entry	Adds address, replaces guard, (ADD2)	Adds address, replaces guard, (ADD2-ID-OFF)	Deletes address and guard (REM1, REM2-DSMM)	Deletes address and guard (REM1-ID-OFF,REM2-DSMM)
1	0	1	0	0	other address and guard	Adds address, adds guard, (ADD3)	Adds address, adds guard, (ADD3)	Nothing	Nothing
1	0	1	0	1	can't happen				
1	0	1	1	0	my guard type but not this client (my guard RR is orphaned)	Nothing	Adds address, replaces guard, (ADD2-	Nothing	Deletes guard (REM1-ID-OFF,REM2-

Other address type	My address type	Other guard type	My guard type	This client	description	What happens on DSMM Add	What happens on DMSS Add with client-id matching off	What happens on DSMM Remove	What happens on DSMM Remove with client-id matching off
							ID-OFF)		DSMM)
1	0	1	1	1	my guard and this client (my guard RR is orphaned)	Adds address, replaces guard, (ADD2)	Adds address, replaces guard, (ADD2-ID-OFF)	Deletes guard (REM1, REM2-DSMM)	Deletes guard (REM1-ID-OFF, REM2-DSMM)
1	1	0	0	0	address only …. static	Nothing	Nothing	Nothing	Nothing
1	1	0	0	1	can't happen				
1	1	0	1	0	my guard type but not this client (something else has managed the other address entry, perhaps with wrong guard?)	Nothing	Replaces address, replaces guard, (ADD2-ID-OFF)	Nothing	
1	1	0	1	1	my guard and this client (something else has managed the other address entry, perhaps with wrong guard?)]	Replaces address, replaces guard, ADD2	Replaces address, replaces guard, (ADD2-ID-OFF)	Deletes address and guard (REM1, REM2-DSMM)	Deletes address and guard (REM1-ID-OFF, REM2-DSMM)
1	1	1	0	0	both addresses, other guard	Nothing UNLESS **ddns-either-guard-is-dynamic** ON: then replace address, add guard, (ADD3-OTHER)	Nothing UNLESS **ddns-either-guard-is-dynamic** ON: then replace address, add guard, (ADD3-OTHER)	Nothing UNLESS **ddns-either-guard-is-dynamic** ON: then delete address (REM1,REM2-DSMM-OTHER)	Nothing UNLESS **ddns-either-guard-is-dynamic** ON: then delete address (REM1-ID-OFF, REM2-DSMM-OTHER)
1	1	1	0	1	can't happen				
1	1	1	1	0	both address and both guards but not this client	Nothing	Replaces address, replaces guard,(ADD2-ID-OFF)	Nothing	Delete address and guard (REM1-ID-OFF,REM2-DSMM)
1	1	1	1	1	both address and both guards, this client	Replaces address, replaces guard,ADD2	Replaces address, replaces guard,(ADD2-ID-OFF)	Delete address and guard (REM1, REM2-DSMM)	Delete address and guard (REM1-ID-OFF, REM2-DSMM)